Rheometer Technical Documentation

Table of Contents

Introduction and Background

Rheometers are lab devices used to measure the viscosity and other dynamic properties (rheology) of fluids. These measurements are typically performed by applying forces and torques to a fluid in a controlled setting then measuring the fluid response.[1] The most common type of rheometer is a shear, also known as a rotational, rheometer. These rheometers measure rheology by applying a torque to a specified geometry in contact with a fluid and recording the stresses and strains.[2] There are also oscillatory rheometers which determine rheology by measuring either the fluid response to a forced vibration or the response of a vibrating system submerged in a fluid.[3] Rheometers are typically expensive (tens of thousands of dollars[4]) due to the sensitivity of the equipment and measurements, and there are no clear options for a more affordable rheometer. Using published research[5] on the dynamics of a submerged spring-mass system with a spherical geometry as a starting point, we aim to produce a low-cost oscillatory rheometer to satisfy the need for an affordable option.

To develop a low-cost rheometer that can adequately predict the viscosity of different fluids, we explore a single degree of freedom oscillating system in the form of a spring-mass system submerged in fluid. We aim to develop a device that can be easily reconfigured to expand the class of fluids for which adequate results may be obtained. That is, the device is designed so that users may easily switch the bobbing mass, the stiffness of the spring, and the external forcing imposed on the mass. Simple variation of these parameters can expand the classes of fluids users may test without deviation from our models presented below.

Project Status

At the time of the creation of this document (December 2020), the rheometer described below can adequately determine the viscosity of Newtonian fluids within about 50% of the actual value, with much more accurate results at lower viscosities. It can report a viscosity for non-Newtonian fluids, but it is unreliable. The only geometry tested so far is a sphere, but the size of the sphere can be tweaked to give more precise results depending on the fluid properties.

---

[1] https://wiki.anton-paar.com/en/basics-of-rheology/#basics-of-rheology
[2] https://www.labcompare.com/Chemical-Analysis-Equipment/602-Rotational-Rheometer/
[3] http://www.mate.tue.nl/~wyss/home/resources/publications/2007/Wyss_GIT_Lab_J_2007.pdf
[4] https://www.ptonline.com/articles/rheometers-which-type-is-right-for-you
[5] Dolfo, G., Vigué, J. & Lhuillier, D. Experimental test of unsteady Stokes' drag force on a sphere. *Exp Fluids* 61, 97 (2020). https://doi.org/10.1007/s00348-020-2936-6

Experimental Methodology and Protocol

To build toward the final goal of determining a fluid's viscosity, we begin by measuring the natural, undamped frequency of the spring-mass system when not submerged in fluid. We can compare this free natural frequency ω_n to a measured damped natural frequency ω_d when the system is submerged in fluid. We model the system with the following general equation of motion with natural (ω_n) and damped (ω_d) frequencies:

$$m\ddot{x} + c\dot{x} + kx = 0 \qquad \omega_n = \sqrt{k/m}, \omega_d = \omega_n\sqrt{1-\xi^2}, \xi = \frac{c}{2m\omega_n}$$

Where m is the mass of the entire system below (but not including) the spring, c is the damping coefficient which is unknown, and k is the spring constant. Although the damping coefficient is unknown, it is related to the drag force experienced by the bobbing mass m. Since the driving frequency ω, the spring constant k, and the bobbing mass m are known and measured quantities, we can estimate the damping coefficient from which we may determine the viscosity of the fluid. To do this, we measure the acceleration of the vibrating system both above and below the spring with accelerometers and use the methods described in **Measurements and Validation** to find the ratio of the bottom (below the spring) acceleration amplitude to the top (above the spring) acceleration amplitude. The ratio of acceleration amplitudes must be found for a range of frequencies (recommended 15 to 75 Hz) to determine the frequency which gives the maximum amplitude ratio (MAR). Once this experimental data is obtained, we then turn to mathematical theory and calculate the MAR for a range of viscosity values given our known values for m and k. We then compare these calculated MARs to our experimentally determined MAR until we reach a viscosity that gives a calculated MAR within 1% of the experimentally determined MAR. Essentially, the experimental method is guess and check where a range of viscosities is swept, and the viscosity that gives a MAR most closely aligned with the experimental results is the final answer.

For Newtonian fluids, it is important that the measured viscosity is close to the reference viscosity by at least the same order magnitude, and that the measured viscosity is nearly shear-rate independent. If the measured viscosity is not shear-rate independent, the fluid is non-Newtonian, and the device currently cannot obtain a reliable measurement of viscosity for non-Newtonian fluids. One of our results presented below is for chocolate syrup which is shear-rate dependent and can have a viscosity between 670 and 5000 cP depending on the shear rate. Our device reports a viscosity of 1380 cP which is within this range, but it has no way of verifying that this is the correct viscosity given the shear rate, so the result is unreliable.

Measurements and Validation

There are two distinct phases of data processing, and sample MATLAB code for both are included in the **Code** section below. Phase one of processing determines the amplitude ratios of the accelerometers, and phase two uses the parameters of the rheometer and the fluid being tested to estimate the fluid viscosity.

In phase one, once you have collected and saved the accelerometer data for a range of frequencies, use a two-signal Fourier transform to find the amplitude ratio between accelerometers at each driving frequency for which data was collected. We provide example code to do this analysis (**First Code**), but it can also be accomplished using any sufficient Fourier transform code or method. The speaker used in our particular setup exhibits nonlinear behavior at low frequencies, so a Fourier transform is required to isolate the acceleration amplitudes at the desired driving frequency ω. Record the ratio of the bottom acceleration amplitude to the top acceleration amplitude for each driving frequency. The frequency which gives the maximum amplitude ratio (MAR) is the damped natural frequency ω_d of the system.

In phase two, plot the amplitude ratios recorded in phase one versus each driving frequency for which data was collected. Using the example code provided (**Third Code**), plug in the mass and radius of the sphere, the density of the liquid, and the spring constant. Then, choose an initial estimate of the viscosity of the fluid and plot the predicted amplitude ratio using the provided code. In the code provided, it is absolutely essential that the initial estimate of the viscosity is lower than the actual viscosity or the code will not work properly, so be extremely conservative in your viscosity estimate. The code will then iterate and automatically adjust the viscosity until it finds a viscosity at which the calculated MAR matches the experimental MAR by within 1%. The code will then return the viscosity value corresponding to this case, which is the final viscosity estimate.

Listed below is a table of results obtained from our testing. The table lists the fluid being tested, our predicted viscosity, and the actual viscosity as measured by a TA Ares G2 Rheometer with a cone and plate geometry. A sample graph showing the acceleration ratio data collected for

100% glycerine with the predicted response found by our MATLAB code is also shown below. After running the code, the viscosity for pure glycerine was estimated to be 940 centipoise, which is only 4.4% higher than the viscosity found by the Ares rheometer (900 centipoise).

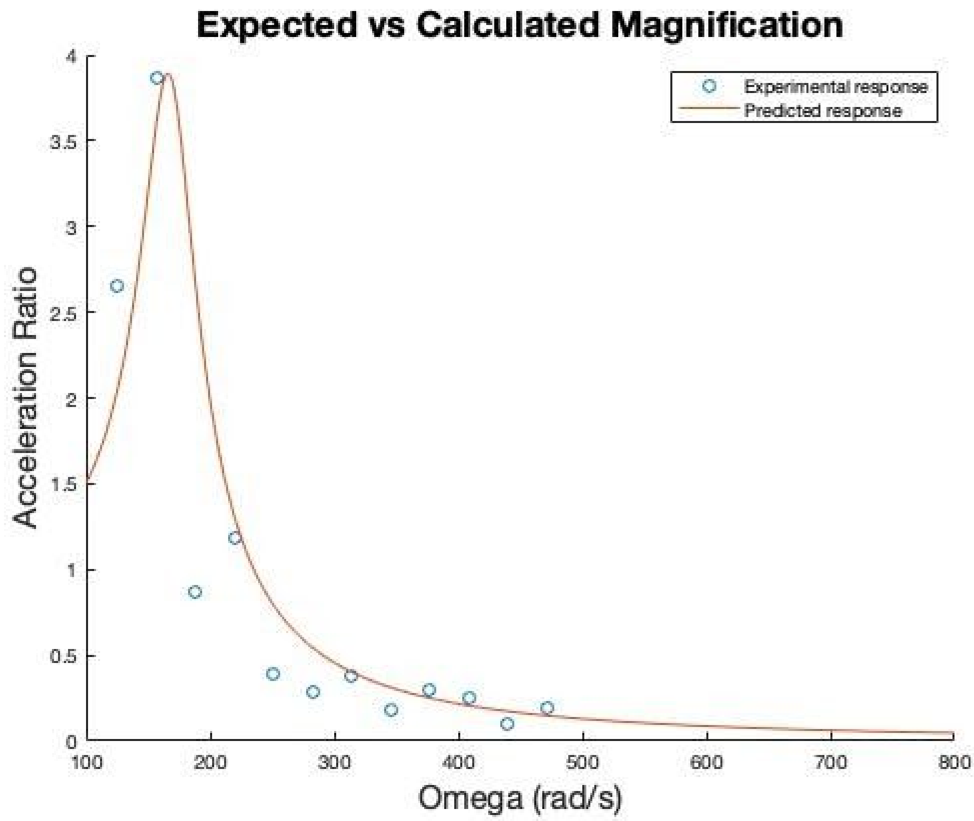

Fig. 1: Predicted versus experimental results for 100% glycerine solution

Fluid	Predicted Viscosity	Actual Viscosity	Percent Difference
100% Glycerine	940 cP	900 cP	4.35%
90% Glycerine/water	240 cP	291 cP	19.2%
Chocolate Syrup	1380 cP	870 cP (at 40 s^{-1})	45.3%
Karo Syrup	4860 cP	8000 cP	48.8%

As is clear from our results above, our device is able to accurately predict the viscosity of Newtonian fluids to within approximately an order of magnitude, with the most accurate results being for fluids with relatively low viscosities.

Theory

Our mathematical model was developed using a paper from Dolfo, Vigué, and Lhuillier about the unsteady drag force on a sphere submerged in a fluid.[6] The paper explores the full unsteady Stokes equation for drag on a sphere:

> The force exerted on the sphere was calculated, in the limit of zero Reynolds number, by Stokes. This force, parallel to the **x**-axis, is the sum of a drag force and an added-mass force (Stokes 1851; Landau and Lifschitz 1959)
>
> $$F_x = -6\pi\eta a\left(1+\frac{a}{\delta}\right)\frac{\mathrm{d}x}{\mathrm{d}t} - \frac{2\pi}{3}\rho a^3\left(1+\frac{9\delta}{2a}\right)\frac{\mathrm{d}^2x}{\mathrm{d}t^2},$$

where η is viscosity, a is the sphere radius, and δ is the viscous penetration depth given by:

$$\delta = \sqrt{2\eta/\rho\omega}$$

where ρ is the fluid density and ω is the driving frequency of the bobbing mass system. The complete mathematics are laid out in the paper cited, but to summarize, the equation for steady Stokes drag on a sphere $F = 6\pi a\eta U$ where U is the velocity of the sphere must be modified in the unsteady case to account for an additional drag component and an added mass component.

To analyze the experimental data, we developed a MATLAB program to read CSV files of accelerometer data and return the corresponding response amplitudes and their ratio at the driving frequency (**Second Code**). We are interested in finding the ratio of the bottom acceleration amplitude to the top acceleration amplitude at only the driving frequency ω, which is why a Fourier transform is necessary to isolate the amplitudes specifically at the frequency ω. The code makes use of the Fast Fourier Transform (FFT) function which converts the accelerometer sinusoids from the time domain into the frequency domain. Once this conversion has been completed, we can extract the amplitudes of the top and bottom accelerometers at our driving frequency and calculate the ratio between them.

We developed a second MATLAB program to calculate the approximate viscosity of the fluid being tested (**Third Code**). This program plots the amplitude ratios at each frequency found with the program described above and also uses the drag force equation from the Dolfo et al. paper to plot a predicted amplitude ratio based on input parameters from the user. The

---

[6] Dolfo, G., Vigué, J. & Lhuillier, D. Experimental test of unsteady Stokes' drag force on a sphere. *Exp Fluids* 61, 97 (2020). https://doi.org/10.1007/s00348-020-2936-6

program requires that the user enter the mass and radius of the sphere, the spring constant, and the density of the fluid being used for data collection. The program also requires that the user enter a guess for the viscosity of the liquid, which is used as a starting value for fitting the predicted amplitude ratios to the experimental ratios. As long as the guess is smaller than the actual value, the program can successfully estimate the viscosity even if they are orders of magnitude apart.

The program uses the constants provided to calculate and plot the amplitude ratio versus frequency of vibration. The original guess for viscosity is used to calculate the predicted amplitude ratios for the range of frequencies tested, and the maximum ratio (peak value) of this plot and the experimental plot are compared. If they differ by more than one percent, the estimated viscosity is increased by 10% of its original value and the calculations described previously are repeated until the desired percent difference is reached. The viscosity that produced a satisfactory percent difference in maximum amplitude ratios is reported. Because this program increases the original guess for viscosity to determine its actual value, it only works if the guess is lower than the actual value. So, the user should always pick an estimate that will certainly be lower than the actual value.

Code

The first code accepts two signals and the driving frequency as inputs, and it returns the amplitudes of both acceleration waveforms and the ratio of the bottom acceleration amplitude to the top acceleration amplitude. This code has been validated for frequencies between 15 and 100 Hz. The second code is example code to call the function defined in the first code and read CSV files of accelerometer data. The third code accepts the mass and radius of the sphere, the spring constant, the fluid density, and an initial viscosity guess as inputs, and it returns the final viscosity. As mentioned previously, the initial viscosity guess must be lower than the actual viscosity for the code to function properly.

First Code

```
function [Ax,Ay,ratio] = fft_two_signals(x,y,z)
% Performs FFT Analysis of the acceleration signals
% and returns their amplitudes Ax, Ay, and ratio.
% Inputs are signals x and y, and driving frequency z.
%% User Inputs
% Sample rate of the Arduino, 1000 corresponds to 115200 Baud
```

```
Fs = 1000;
%% Processing Code
% No code here needs to be changed by the user.
x=x-mean(x);
y=y-mean(y);
T = 1/Fs;
L=length(x);
X = fft(x);
Y = fft(y);
f = Fs*(0:(L/2))/L;
P2X = abs(X/L);
P1X = P2X(1:floor(L/2)+1);
P1X(2:end-1) = 2*P1X(2:end-1);
P2Y = abs(Y/L);
P1Y = P2Y(1:floor(L/2)+1);
P1Y(2:end-1) = 2*P1Y(2:end-1);
% Refine the range to driving frequency so that the
% max function gives the amplitude at the frequency
% we are interested in:
P1Y=P1Y(round(z/f(2))-25:round(z/f(2))+25);
P1X=P1X(round(z/f(2))-25:round(z/f(2))+25);
[Ay,I2]= max(P1Y);
f2=f(I2);
[Ax,I1]= max(P1X);
f1=f(I1);
ratio = Ay/Ax;
% Code used for plotting, visualization
figure
plot(f(round(z/f(2))-25:round(z/f(2))+25),P1X)
xlim([z-5,z+5])
hold on
plot(f(round(z/f(2))-25:round(z/f(2))+25),P1Y)
% Print out answers
fprintf('The amplitude of the base is: %5.4f\n',Ax)
fprintf('The amplitude of the mass is: %5.4f\n',Ay)
fprintf('The magnification ratio is: %5.4f\n',ratio)
end
```

Second Code

```
% Create a matrix of acceleration ratios for the two accelerometers used
%   in the rheometer for each frequency tested

%% User instructions for processor
% Name your csv files with accelerometer data "15hz.csv" for 15Hz, etc
  % This code is written for frequencies of 15 to 75 Hz with an interval
  % of 5 HZ

% Change the purple text below to the name of the folder your data is in:
  addpath('medium/glycerin_090')
% in this case, 'medium' is the size of the sphere and
%   'glycerin_100' is the name of the folder containing the data

% If you would like to test different frequencies, simply copy and
% paste the block of code below, filling the frequency for each [freq]

%      M[freq] = readmatrix('[freq]hz.csv');
%      top[freq]=M[freq](:,1);
%      botx=M[freq](:,2);
%      [AT[freq],AB[freq],ratio[freq]]=fft_two_signals(top[freq],bot[freq],15);

% You must also edit the matrix in line 96 to include ratio[freq]

% Make sure the frequencies in ratio_mat are in ascending order
```

```
%% processing code
% calculate ratio for each frequency where data was collected

M15 = readmatrix('15hz.csv');
top15=M15(:,1);
bot15=M15(:,2);
[AT15,AB15,ratio15]=fft_two_signals(top15,bot15,15);

M20 = readmatrix('20hz.csv');
top20=M20(:,1);
bot20=M20(:,2);
[AT20,AB20,ratio20]=fft_two_signals(top20,bot20,20);

M25 = readmatrix('25hz.csv');
top25=M25(:,1);
bot25=M25(:,2);
[AT25,AB25,ratio25]=fft_two_signals(top25,bot25,25);

M30 = readmatrix('30hz.csv');
top30=M30(:,1);
bot30=M30(:,2);
[AT30,AB30,ratio30]=fft_two_signals(top30,bot30,30);

M35 = readmatrix('35hz.csv');
top35=M35(:,1);
bot35=M35(:,2);
[AT35,AB35,ratio35]=fft_two_signals(top35,bot35,35);

M40 = readmatrix('40hz.csv');
top40=M40(:,1);
bot40=M40(:,2);
[AT40,AB40,ratio40]=fft_two_signals(top40,bot40,40);

M45 = readmatrix('45hz.csv');
top45=M45(:,1);
bot45=M45(:,2);
[AT45,AB45,ratio45]=fft_two_signals(top45,bot45,45);

M50 = readmatrix('50hz.csv');
top50=M50(:,1);
bot50=M50(:,2);
[AT50,AB50,ratio50]=fft_two_signals(top50,bot50,50);

M55 = readmatrix('55hz.csv');
top55=M55(:,1);
bot55=M55(:,2);
[AT55,AB55,ratio55]=fft_two_signals(top55,bot55,55);

M60 = readmatrix('60hz.csv');
top60=M60(:,1);
bot60=M60(:,2);
[AT60,AB60,ratio60]=fft_two_signals(top60,bot60,60);

M65 = readmatrix('65hz.csv');
top65=M65(:,1);
bot65=M65(:,2);
[AT65,AB65,ratio65]=fft_two_signals(top65,bot65,65);

M70 = readmatrix('70hz.csv');
top70=M70(:,1);
bot70=M70(:,2);
[AT70,AB70,ratio70]=fft_two_signals(top70,bot70,70);

M75 = readmatrix('75hz.csv');
top75=M75(:,1);
```

```
bot75=M75(:,2);
[AT75,AB75,ratio75]=fft_two_signals(top75,bot75,75);

% put ratios for each freq in one matrix
ratio_mat = [ratio15 ratio20 ratio25 ratio30 ratio35 ratio40 ratio45...
    ratio50 ratio55 ratio60 ratio65 ratio70 ratio75];
```

Third Code

```
% Model rheometer and predict behavior for different liquids
    % plot experimental acceleration ratios
    % use plot to find viscosity that matches experimental data

%% User instructions for rheometer
    % 1. run accelprocessor_foruser for your accelerometer data
    % 2. enter the mass of the system below the spring below
    mass = 29.63*10^-3; % kg
    % 3. enter the radius of the sphere below
    radius = 0.0127; %m
    % 0.015; large
    % 0.0127 medium
    % 0.009 small
    % 4. enter the spring constant of the spring
    k = 1050.76; % Nm
    % 5. enter the density of the fluid
    density = 1231; % kg/m^3
    % 6. enter your guess for the viscosity of the fluid
    eta = 100; % cP
    % 7. enter the ratio_mat data obtained from accelprocessor_foruser.m
    ratio_mat = [];
%% processing code

% DEFINE CONSTANTS IN FORCE EQUATION (FROM DOLFO PAPER)
% convert viscosity from cP to Pa*s
eta_guess = eta*10^-3;
% create matrix for range of omega values
omega = linspace(80,500,800);
% viscous penetration depth (m)
delta = sqrt(2*eta_guess./(density.*omega));
% calculate damping coefficient, Nm/s
c = -6*pi*eta_guess*radius.*(1+radius./delta);
% natural frequency, rad/s
omega_n = sqrt(k/mass);
% calculate viscous damping factor
zeta = c./(2*mass*omega_n);
% ratio of frequency to natural frequency
r = omega/omega_n;
% set up matrix to store calculated acceleration ratios
Accel_ratio = [];
% calculate acceleration ratios at every frequency
for i = 1:length(omega)
    Accel_ratio(i) = 1/sqrt((1-r(i)^2)^2+(2*zeta(i)*r(i))^2);
end

% set up matrix of frequencies at which experimental data was collected
omega_exp = 15:5:75;
omega_exp = 2*pi*omega_exp;

%% calculate best approximation of viscosity
% comment this out if you want to brute force the approximation

% find maximum acceleration ratio in exp. data
max_A = max(ratio_mat);
% find maximum acceleration ratio in estimated calculations
A_approx = max(Accel_ratio);
```

```
% find % difference between maximum accel ratios
percent_dif_w = abs((max_A-A_approx))/max_A;

% define the increment that the approximate viscosity should increase by
%   to be 10% of its estimated value
increment = 0.1*eta_guess; % this can be adjust by changing the fraction

% continue to increase estimated viscosity until max amplitude give %
%   difference of less than 1
while percent_dif_w > 0.01
    eta_guess = eta_guess + increment; % increase estimate
    % recalculate constants
    delta = sqrt(2*eta_guess./(density.*omega));
    c = -6*pi*eta_guess*radius.*(1+radius./delta);
    zeta = c./(2*mass*omega_n);
    for i = 1:length(omega)
        Accel_ratio(i) = 1/sqrt((1-r(i)^2)^2+(2*zeta(i)*r(i))^2);
    end
    [A_approx, omegan_approx] = max(Accel_ratio); % recalculate max accel ratio
    percent_dif_w = abs((max_A-A_approx))/max_A; % recalculate % dif
end


%% Plot experimental and calculated acceleration ratios
figure
hold on

% plot experimental and calculated accel ratios vs frequency
scatter(omega_exp, ratio_mat)
plot(omega, Accel_ratio)

xlabel('Omega (rad/s)','fontsize', 16)
ylabel('Acceleration Ratio','fontsize', 16)
title('Expected vs Calculated Magnification','fontsize',18)
xlim([80 500])
%ylim([0 10])
legend('Experimental response','Predicted response')

% print calculated viscosity
visc = ['Viscosity = ', num2str(eta_guess*10^3)];
disp(visc)
```

Diagrams and Visuals

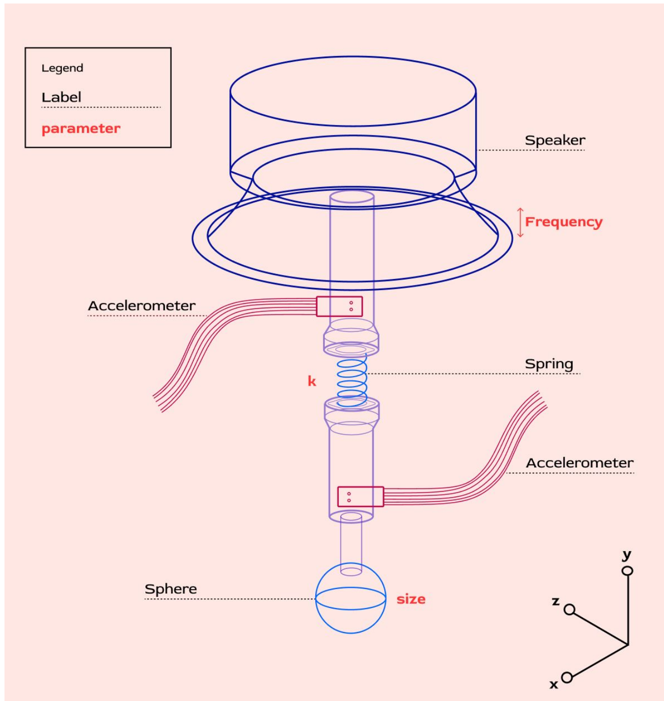

Fig. 2: Schematic diagram of the Rheometer device

Figure 2 shows the components of the vibrating mass system and indicates what setup parameters we varied in our testing. These parameters were frequency, spring constant, and the size of our sphere. Of these, we explored springs with k constants of 6 (lbs/in) and 10 (lbs/in), sphere radii of 1.5 cm and 0.9 cm, and frequencies ranging from 15 Hz to 75 Hz.

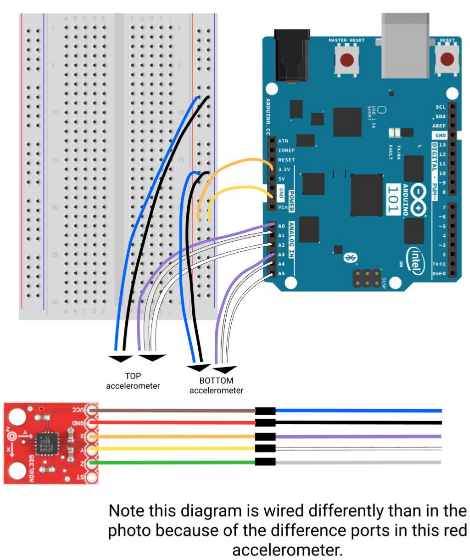

Fig. 3: Wiring Diagram

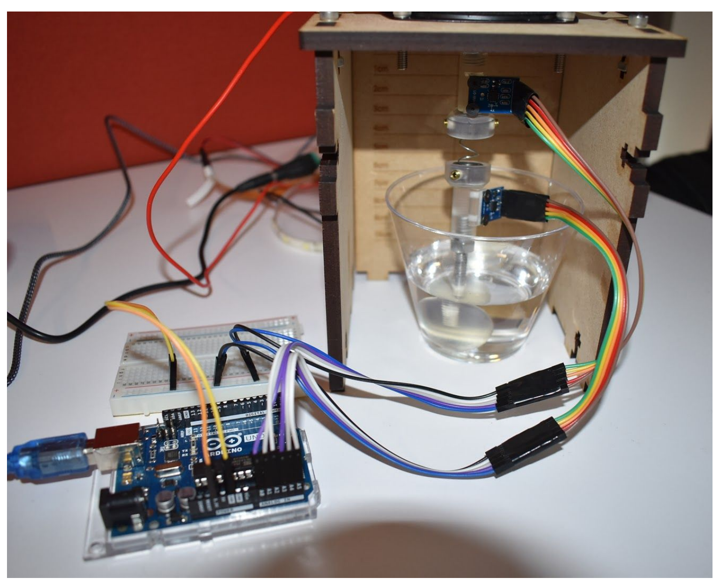

Fig. 4: Final Setup

Further diagrams, photos, materials needed and instructions on device assembly are available in our instructables post.

Future Steps

The way that this experiment is designed requires that the fluid is Newtonian. That is because as the driving frequency changes, the shear rate also changes. The data collection process requires that viscosity be independent of driving frequency and, as a consequence, must also be independent of shear rate. It may be useful going forward to consider a set of experiments and dynamic models that account for the shear-rate dependence of non-Newtonian fluids in order to characterize the power-law behavior.

Furthermore, the current computer program requires that the user input an initial guess for viscosity that is lower than the error-reducing value found through the sweep. Future work could include a more sophisticated search algorithm for finding an error-reducing viscosity that can search in multiple directions more efficiently than a fixed resolution sweep, and could take into account the entire shape of the response curve rather than just the maximum amplitude ratio.